Implementarea filtrelor digitale FIR în forma lattice

Laborator 7, PSS

Table of contents

1 Obiectiv

Familiarizarea studenților cu formele de implementare tip *lattice* folosite la implementarea filtrelor de tip FIR

2 Noțiuni teoretice

Implementarea în formă *lattice* a unui filtru FIR de ordin 3:

Ecuații:

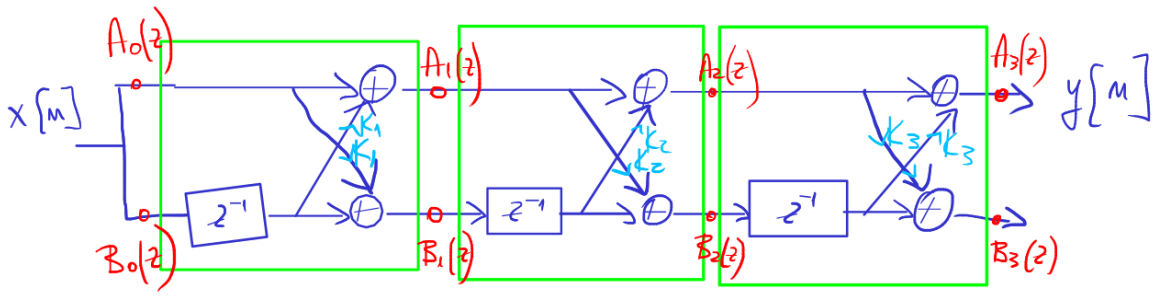

Figure 1: Forma lattice, ordin 3

$$
\begin{aligned}
A_0(z) &= B_0(z) = 1 \\
A_m(z) &= A_{m-1}(z) + K_m \cdot z^{-1} \cdot B_{m-1}(z) \\
A_{m-1}(z) &= \frac{A_m(z) - K_m \cdot B_m(z)}{1 - K_m^2} \\
B_m(z) &= z^{-m} B_m(z^{-1}) = \text{ similar cu } A_m(z), \text{ cu coeficienții în ordine inversă}
\end{aligned}
$$

Aceste ecuații permit calcularea coeficienților de reflexie K_m din $H(z)$, sau calcularea $H(z)$ dacă se cunosc K_m.

3 Exerciții teoretice

1. Determinați coeficienții filtrului FIR în forma directă dacă se cunosc coeficienții de reflexie ai structurii *lattice*: $K_1 = \frac{1}{2}$, $K_2 = 0.6$, $K_3 = -0,7$, $K_1 = \frac{1}{3}$.
2. Determinați coeficienții structurii *lattice* pentru un filtru FIR cu funcția de sistem:

$$H(z) = 1 + \frac{2}{5}z^{-1} + \frac{7}{20}z^{-2} + \frac{1}{2}z^{-3}$$

4 Exerciții practice

1. În Matlab, utilizați utilitarul `fdatool` pentru a proiecta unul din filtrele următoare:
   a. Un filtru trece-jos FIR de ordin 5, de tip echiriplu, cu frecvența de tăiere de 5kHz la o frecvență de eșantionare de 44.1kHz;
   b. Un filtru trece-sus FIR de ordin 5, de tip echiriplu, cu frecvența de tăiere de 1kHz la o frecvență de eșantionare de 44.1kHz;
   c. Un filtru trece-bandă FIR de ordin 5, de tip echiriplu, cu banda de trecere între 700Hz si 4kHz la o frecvență de eșantionare de 44.1kHz.

2. Exportați coeficienții filtrului de mai sus în Workspace-ul Matlab, cu numele `Num`.

   Utilizați funcția `tf2latc()` pentru a converti coeficienții din vectorul `Num` (corespunzători formelor directe) în coeficienț ai formei *lattice*. Denumiți vectorul rezultat K și afișați-l.

   Utilizați apoi și funcța inversă `latc2tf()` pentru a converti coeficienții din forma *lattice* `K` înapoi în coeficienți ai formei directe, si verificați că se obțin aceleași valori ca în `Num`.

   Observația 1:

   - E posibil să aveți o eroare la funcția `tf2latc()`, de forma:

     ```
     Error using levdown
     At least one of the reflection coefficients is equal to one.
     The algorithm fails for this case.
     ```

     Acest lucru se întâmplă pentru că vectorul `Num` este simetric (primul coeficient este egal cu ultimul, etc) (filtru de fază liniară).

     Pentru a evita eroarea, trebuie să stricăm această simetrie, de exemplu adăugând un mic ϵ la primul coeficient:

     ```
     Num(1) = Num(1) + 0.0000001
     ```

   Observația 2:

   - Utilizând `latc2tf()` nu veți obține exact coeficienții inițiali din `Num`, ci coeficienții raportați la prima valoare, `Num / Num(1)`. Acest lucru se întâmplă pentru că forma de implementare *lattice* necesită ca primul coeficient din $H(z)$ să fie neapărat de valoare 1.

     Așadat, coeficienții din vectorul `K` nu implementează de fapt filtrul `Num`, ci filtrul `Num / Num(1)`. Asta înseamnă că în schema Simulink trebuie ca semnalul de intrare să fie trecut printr-un Gain suplimentar înainte de a intra în schema *lattice*, de valoare `Num(1)` (adică b_0).

3. În mediul Simulink, realizați implementarea FIR filtrului de mai sus în forma *lattice*.

   a. Implementați schema, punând în Gain-uri coeficienții din vectorul `K` (`K(1)`, `K(2)`, etc)
   b. Puneți la ieșire un bloc de vizualizare ("Scope") și afișati răspunsul la impuls (intrarea "Discrete Impulse") și răspunsul la treapta unitate (intrarea "Step").

4. În Matlab, creați o funcție pentru a filtra un vector `x` cu un filtru FIR pentru care se cunosc coeficienții *lattice* `K`, obținînd vectorul de ieșire `y`.

   Găsiți o metodă de a calcula ieșirea `y[n]` la un moment oarecare `n`, pe baza schemei. Cu alte cuvinte, dacă avem schema, cum putem scrie niște ecuații sau linii de cod pentru a o implementa?

   Puteți utiliza următorul template orientativ:

```
function y = filter_lattice_FIR(K, x)
% Filter the vector x with a lattice FIR filter with coefficients K

ord = length(K);

% Write code here:
...

end
```

5. Testați funcția, apelând-o cu coeficienții `K` ai filtrului proiectat anterior, pentru semnalul de intrare `x` de forma:

   a. un vector cu un 1 urmat de 19 zerouri (impuls unitate)
   b. un vector cu 20 de 1 (treapta unitate)

   Afișati grafic vectorii obținuți la ieșire, cu funcția `stem()`.

5 Întrebări finale

1. TBD